



Spring Data JPA系列:继承的方法

上一节讲了spring-data-jpa的基本配置，这一节主要是讲自定义Repository继承了JpaRepository相关的一些方法，包括基本的增删改查方法。

1、方法列表

- 1)、从PagingAndSortingRepository继承的findAll方法
- 2)、从CrudRepository继承的count, delete, deleteAll, exists, findOne, save等方法
- 3)、从QueryByExampleExecutor继承的count, exists, findAll, findOne**

2、demo示例

因为比较简单，这里只是简单写几个简单的方法进行测试，这里为了简单方便起见，我讲测试的方法放到Controller内——Customercontroller。

1)、save方法，初始化，保存Customer数据**

```
@Controller
@RequestMapping("/customer")
public class CustomerController {
    @Autowired
    private CustomerRepository repository;

    /**
     * 初始化数据
     */
    @RequestMapping("/index")
    public void index() {
        // save a couple of customers
        repository.save(new Customer("Jack", "Bauer"));
        repository.save(new Customer("Chloe", "O'Brian"));
        repository.save(new Customer("Kim", "Bauer"));
        repository.save(new Customer("David", "Palmer"));
        repository.save(new Customer("Michelle", "Dessler"));
        repository.save(new Customer("Bauer", "Dessler"));
    }
}
```

这里调用了接口继承的save方法。

2)、findAll方法，查找出完成初始化的数据

```
/**
 * 查询所有
 */
@RequestMapping("/findAll")
public void findAll(){
    List<Customer> result = repository.findAll();
    for (Customer customer:result){
        System.out.println(customer.toString());
    }
    System.out.println("-----")
}
```

3)、findOne方法，根据long型Id为参数，返回对应的人

Spring Data

阅读 478 评论 0 喜欢 0

喜欢



SpringForAll

文章 79 问答 6 粉丝 236

关注

```
/**
 * 查询ID为1的数据
 */
@RequestMapping("/findOne")
public void findOne(){
    Customer result = repository.findOne(1L);
    if(result!=null){
        System.out.println(result.toString());
    }
    System.out.println("-----")
}
```

4)、delete方法, 根据Id删除指定数据, 也可以用批量删除方法

```
/**
 * 查询ID为1的数据
 */
@RequestMapping("/delete")
public void delete(){

    System.out.println("删除前数据: ");
    List<Customer> customers = repository.findAll();
    for (Customer customer:customers){
        System.out.println(customer.toString());
    }

    System.out.println("删除ID=3数据: ");
    repository.delete(3L);

    System.out.println("删除后数据: ");
    customers = repository.findAll();
    for (Customer customer:customers){
        System.out.println(customer.toString());
    }
    System.out.println("-----")
}
```

5)、其他还有很多, 用法方法, 大同小异, 只是在功能上不完全重叠。

是不是感觉有JPA真心方便, 不用谢mapper文件, 不用各种配置, 甚至不用定义基本的增删改查方法, 写个类似DAO的类, 继承一下JpaRepository或者CrudRepository, 当程序运行的时候, 会创建一个虚拟的实训类, 然后执行相关逻辑。

这一节就到这里, 下一节讲通过方法名来实现一些简单查询, 有一定的局限性, 但是可以学习一下这种操作, 以备不时之需。

参考:

官方文档:<https://docs.spring.io/spring-data/jpa/docs/current/reference/html>

API官方文档:<http://docs.spring.io/spring-data/data-jpa/docs/current/api/>

JPQL文档:<http://www.blogjava.net/calmJava/archive/2011/04/01/347450.html>

DEMO示例: <https://github.com/icnws/spring-data-jpa-demo>

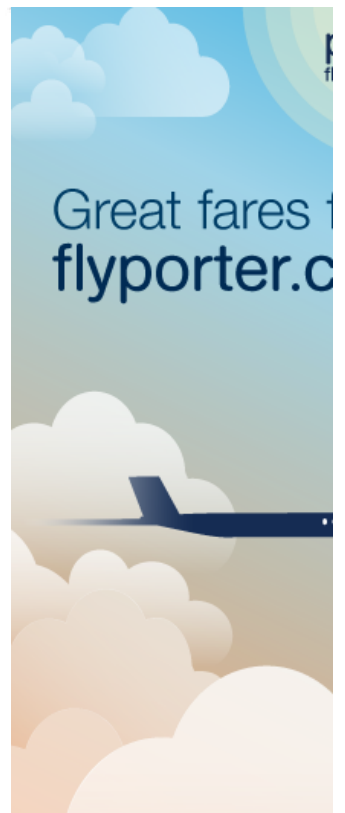
声明:

转载自: <http://www.icnws.com/2017/spring-data-jpa-inherit-functions/>

标签:

HTTP

Spring Data



Some conditions apply.

Sea

▼ 评论